REMARKS

This is a full and timely response to the outstanding non-final Office Action mailed April 21, 2004. Reconsideration and allowance of the application and pending claims are respectfully requested.

Claim Rejections - 35 U.S.C. § 103(a)

Claims 1-5, 7-11, and 13-18 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Pekowski (U.S. Pat. No. 5,946,486) in view of Han (U.S. Pat. No. 6,718,334). Applicant respectfully traverses this rejection.

As is indicated above, Applicant has amended each independent claim and various dependent claims. In view of these amendments, Applicant respectfully submits that the rejection is moot as having been drawn against the claims in their previous form. Applicant discusses Pekowski and Han in the following, however, for the Examiner's consideration.

A. The Pekowski System

Pekowski discloses an apparatus and method for tracing entries to or exits from a target dynamic link library (DLL). As is described by Pekowski, a shadow DLL is generated that acts as an interceptor of all calls that are made to the target DLL. Pekowski, column 4, line 55 to column 5, line 5. Therefore, each call to the target DLL and return from the DLL passes through the shadow DLL such that the shadow DLL can trace events occurring upon entry to or exit from the target DLL. Id.

This manner of operation is made possible because the shadow DLL "looks just like" the target DLL. Id. at column 5, lines 32-41. Therefore, the target DLL has "the same DLL name and the same name and number of external interfaces or exports as the

original." Id. With such a configuration, "[r]enaming the original DLL requires both the external and internal DLL names to be changed."

In view of the above, Pekowski discloses a system that is similar to that identified as prior art in Applicant's own disclosure (see pages 10-11; Figure 2C). Therefore, the Pekowski system incorporates disadvantages identified in Applicant's specification. One such advantage is described on page 11, lines 7-10:

> One of the disadvantages with this architecture is that the intercept library 54 must look exactly like the API library 55. This is because the intercept library 54 completely replaces the API library 55 from the application program 53 perspective.

Applicant's claimed system was developed, at least in part, to avoid such a disadvantage. In particular, unlike the Pekowski system, Applicant's system, in at least some embodiments, does not include an intercept library that completely replaces the API library. Instead, the target library (i.e., the API library) actually receives the event, determines whether an intercept library is enabled to process the event, and, if so, provides the event to the intercept library using a generic interception communication interface having an intercept event send handler. Therefore, no intercept library sits between the application program that generates an event and the API in Applicant's system.

Such an arrangement is, for example, recited in claim 1, which provides (emphasis added):

1. A method for intercepting an event, the method comprising:

generating an event with an application program;

calling an application program interface to process the event;

receiving the event with the application program interface;

automatically ***determining*** without prompting from a user ***if an intercept library is enabled to process the event***;

if the intercept library is enabled to process the event, automatically ***transmitting the event from the application program interface to a generic interception communication interface having at least one intercept event send handler***, the generic interception communication interface maintaining communication between the application program interface and the intercept library;

transmitting the event from the generic interface communication interface to the intercept library with the at least one send handler;

determining if the event is to be processed by the intercept library; and

if the event is to be processed by the intercept library, processing the event with the intercept library.

From the above, it is clear that the Pekowski system fails to disclose or suggest various features of Applicant's claimed inventions.

B. The Han System

As is noted above, the Pekowski reference is deficient as to several features of Applicant's claimed inventions. The Han reference has now been cited to support the Pekowski reference in rejecting Applicant's claims.

Han discloses a computer implemented document an image management (DIM) system. As is described by Han, the DIM system works with a legacy application (e.g., in the Windows™ environment) to save, retrieve, index, process, print, manage, or other process data. Han, column 6, lines 4-10. Although the DIM system is described as monitoring for "triggering events," the DIM system is user-activated (i.e., manually activated). Han, column 6, lines 18-23. In particular, the DIM system is activated by the user by selecting "hot-keys" or by inputting particular mouse clicks. Id.

Han states that "the goal of the invention is to capture the index string that the user's application displays on the screen" using the DIM system. Id. at column 6, lines 43-58. In operation, the DIM system is triggered by the user to intercept a call from an application program to a Windows™ Graphical Devices Interface (GDI). Id. The DIM system, in providing its management services, "can simply instruct the Windows(R) GDI program as to the screen position where the index should be displayed." Id.

In view of the above, application program calls to the program interface (the "GDI") are intercepted by the interceptor (the DIM system) when triggered to do so by the user. Then, the DIM system simply instructs the program interface (GDI) how to process the event (i.e., where to position the captured index string). Accordingly, with reference to claim 1, the Han system is not configured for "receiving the event with the application program interface" because the DIM system intercepts the event before it reaches the GDI. In addition, the Han system is not configured for "automatically determining without prompting from a user if an intercept library is enabled to process

the event" given that Han's DIM system is manually triggered by the user (e.g., by pressing a "hot-key"). Moreover, nothing in the DIM system determines whether an intercept library is enabled. Instead, the DIM system simply intercepts the call at the request of the user.

Han is deficient for other reasons. With further reference to claim 1, for example, Han does not describe "automatically transmitting the event from the application program interface to a generic interception communication interface having at least one intercept event send handler, the generic interception communication interface maintaining communication between the application program interface and the intercept library". First, Han's DIM system intercepts the application program call and, therefore, Han does not teach an API that transmits the event to an intercept library. Second, Han is silent as to the use of a "generic interception communication interface" in completing that transmission. Again, the call from the application program in the Han system is intercepted by the DIM system, which then instructs the application interface (the "GDI") what action to take. Therefore, no component is receiving an event from an API and using another interface to transmit it to an intercept library in the Han system. Third, Han says nothing about an intercept "library". Instead, Han only discloses an application program (e.g., a Windows™ application), a first interface (i.e., the GDI), and the DIM system. Nowhere is the DIM system described as being a library.

As a further matter, Applicant notes that Han does not disclose or suggest "transmitting the event from the generic interface communication interface to the intercept library with the at least one send handler", "determining if the event is to be processed by the intercept library", or "if the event is to be processed by the intercept library, processing the event with the intercept library". As to the last two quotations,

Han does not teach or suggest a library that processes an event. Instead, as noted previously, Han teaches a DIM system that "instructs" the Windows™ GDI program. See Han, column 6, lines 55-58.

Applicant notes that the other claims under consideration contain similar limitations that are also not found in Han.

Although Han is cited in combination with Pekowski, Applicant notes that the deficiencies of both references results in a combination that still does not teach or suggest Applicant's inventions and, therefore, does not render Applicant's claims obvious. Applicant therefore respectfully requests that the rejection be withdrawn.

New Claims

As identified above, claims 21-26 have been added into the application through this response. Applicant respectfully submits that these new claims describe an invention novel and unobvious in view of the prior art of record and, therefore, respectfully requests that these claims be held to be allowable.

CONCLUSION

Applicant respectfully submits that Applicant's pending claims are in condition for allowance. Favorable reconsideration and allowance of the present application and all pending claims are hereby courteously requested. If, in the opinion of the Examiner, a telephonic conference would expedite the examination of this matter, the Examiner is invited to call the undersigned attorney at (770) 933-9500.

Respectfully submitted,

David R. Risley
Registration No. 39,345

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail, postage prepaid, in an envelope addressed to: Assistant Commissioner for Patents, Alexandria, Virginia 22313-1450, on

July 1, 2004

Mary Meegan

Signature